TrailsCommunity

Rapport de projet - Assitant Offroad

BELLANGER Stéphen MONNIER Ysée DESHORS Yann

Novembre 2016 - Decembre 2016

Table des matières

Table des figures

Première partie

Spécification des exigences logicielles

Chapitre 1

Introduction

Cette section fournit une vue d'ensemble de tout ce qui est inclus dans ce document SRS. De plus, l'objectif de ce document est détaillé et une liste de définitions est fournie.

1.1 Objectifs

L'objectif de ce document est de fournir une description détaillée des conditions requises pour l'application mobile TrailsCommunity. Il illustrera le but et la description complète du développement du système. Il expliquera également les contraintes du système, l'interface et les interactions avec les applications externes. Ce document est principalement destiné à être proposé à un client pour son approbation. C'est aussi une référence pour le développement de la première version du système pour l'équipe de développement.

1.2 Portée

TrailsCommunity est une application mobile GPS qui permet aux utilisateurs de connaitre en temps réel le positionnement des autres participants d'une activité en pleine air. L'ensemble des activités sont regroupés en type qui seront limité à une liste pré-défini. L'application doit être gratuite, simple d'utilisation et ergonomique. L'organisateur de l'activité devra fournir deux adresses. Une pour le point de départ de l'application et la deuxième pour le point d'arrivé. La création des sessions se fera directement sur l'application mobile. En outre, le logiciel a besoin à la fois d'Internet et d'une connexion GPS pour récupérer et afficher les résultats. Tous les systèmes d'informations sont conservés dans une base de données, qui se trouve sur un serveur web. De plus une base de données interne sera créer pour pouvoir stocker différentes données relatives aux sessions lorsque l'utilisateur utilise l'application en hors-ligne. Le logiciel intéragit également avec l'API Google Map qui permettra d'afficher et de tracer les différentes parcours des utilisateurs. De plus, le téléphone doit aussi être composé d'un GPS pour pouvoir localiser l'utilisateur à tout moment. L'application a également la capacité de représenter à la fois des informations sommaires et détaillées des différentes statistiques de l'utilisateur courant mais aussi des autres participants de la session en cours.

1.3 Définitions

session une session est une activité disponible dans l'application

visiteur personne n'étant pas identifier

utilisateur personne identifié à l'application. Il possède les choix de créer ou de rejoindre une session

organisateur utilisateur ayant créé une session et pouvant la gérer

participant utilisateur qui a rejoint une session

waypoint désigne un point de la route a atteindre ou doit avoir lieu un changement de cap

1.4 Références

The Institute of Electrical and Electronic Engineer NY USA IEEE Recommented Practice for SRS, `http://www.utdallas.edu/~chung/RE/IEEE830-1993.pdf`

Chalmers IEEE standards - SRS Example, 2010, `http://www.cse.chalmers.se/~feldt/courses/reqeng/examples/srs_example_2010_group2.pdf`

Droid5 Informatics Pvt Ltd androidhive, 2016, `http://www.androidhive.info`

Oracle Java 8 Documentation, 2016, `https://docs.oracle.com/javase/8/docs/api/`

Google Android Documentation, 2016, `https://developer.android.com/guide`

James Britt Ruby Doc, 2016, `http://ruby-doc.org`

Rails Community Ruby On Rails Documentation, 2016, `http://rubyonrails.org`

1.5 Aperçu

Différentes techniques de spécification sont utilisées afin de préciser les exigences plus précisément pour différents publics. Cependant, le lecteur principal sera le correcteur du projet.

Le reste de ce document comprend quatres chapitres. Le second fournit une vue d'ensemble de la fonctionnalité du système et de l'intéraction de l'application avec d'autres systèmes. En outre, le chapitre mentionne également les contraintes du systèmes et les hypothèses sur le produit. Le troisième chapitre fournit la conception et la description du serveur. Il regroupe l'ensemble des conventions, des codes statut ainsi que l'ensemble de ses ressources. Le quatrième chapitre regroupe l'ensemble des annexes du projet. Ce sont en globalité des diagrammes UML. Ils permettent soit une meilleurs finesse de compréhension du système soit, une meilleurs spécification métier du serveur.

Chapitre 2

Description générale

2.1 Perspective du produit

Ce système ce compose de deux parties : une application mobile et un serveur web. L'application Android sera utilisé pour la gestion des sessions. Elle permettra d'afficher en fonction d'une session sélectionner, sur une carte Google map les différentes positions des participants.

Le serveur sera tant qu'à lui capable de gérer la récolte et la gestion des données envoyer par l'application. De plus, un mode hors-ligne sera disponible sur l'application. En effet, elle basculera sur une base de données interne qui stockera l'ensemble des données. Lorsque l'utilisateur se reconnectera à internet. L'ensemble des informations stockées en interne seront envoyé au serveur.

L'application mobile devra communiquer avec un GPS qui se situe au sein du mobile. Le GPS fournira les emplacements de l'utilisateur. Pour ce qui est des autres participants, leurs coordonnées seront stockées dans la base de données du serveur. De plus, des statistiques sur l'utilisateur courant ainsi qu'un chat seront disponible pour permettre au différents membres de pouvoir communiquer ensemble.

De manière transparente comme il s'agit d'un produit axé sur la récolte de données, il lui faudra une base de données. La communication avec la base de données se passera via internet. Cependant, comme expliqué précédemment, l'application possèdera une base de données interne pour gérer le mode hors-ligne.

2.2 Fonctions du produit

Avec l'application mobile, les utilisateurs pourront créer un compte utilisateur. Cette première étape va permettre de collecter différentes informations primordiales sur les utilisateurs.

Après la réalisation de cette première étape, et la connexion de ceux-ci, les utilisateurs pourront sélectionner des sessions. Le résultat sera basé sur les sessions que l'utilisateur sélectionne. Il existe 3 types de sessions. Les sessions actives, elle sont encore réalisable et ne sont pas encore clôturées.

Cela veut donc dire que leurs dates de fin ne sont pas dépassées. Dans ces sessions, un chat en temps réel est disponible permettant de communiquer avec tout les participants.

Les sessions clôturé dont l'utilisateur à participé. Elle ce classe dans la section historique qui permet à l'utilisateur courant de pouvoir revisualisé ses anciennes statistiques.

Enfin les sessions dont l'utilisateur est le créateur. On l'appelle l'organisateur dans les acteurs. Il a donc les droits de modifications sur celle-ci.

L'application va permet après sélection d'une session, d'afficher une carte Google map ainsi que ensemble de point permetant de décrire le déplacement des différents utilisateurs actif de la session. De plus, les sessions pourront être directement créée depuis l'application.

2.3 Caractéristique de l'utilisateur

Il existe quatres types d'acteurs qui interagissent avec le système : le visiteur, l'utilisateur, le participant et l'organisateur. Chacun de ces quatres a une utilisation différente du système affin que chacun ait leurs propres exigences.

Les visiteurs de l'application mobile ne peuvent utiliser TrailsCommunity si il ne sont pas connecter. Cela signifie que le visiteur doit être en mesure de pouvoir créer un compte utilisateur et ainsi de pouvoir se connecter.

Les visiteurs connectés peuvent alors visualiser l'ensemble des sessions actives mais aussi leur historique de session ainsi que les sessions dont ils sont organisateurs. De plus, ils peuvent modifier leurs coordonnées personnelles.

Les participants sont des acteurs qui ont rejoint une session active, ils peuvent alors ajouter un waypoint et le partager à l'ensemble des participants. Ils peuvent aussi communiquer via un chat en temps réel. De plus, ils peuvent consulter leurs statistiques en cours.

Enfin, les organisateurs sont des utilisateurs qui ont créée une nouvelle session. En effet, ce sont eux dont l'impulsion est venu pour organiser une activités. Ce sont eux qui vont pouvoir partager le mot de passe de la session pour pouvoir la rejoindre.

2.4 Contraintes

L'application mobile est limitée au système de navigation GPS du téléphone portable. Comme il existe plusieurs système de fabricants de GPS, la précision n'est pas la même pour chacun d'entre eux. En outre, il peut y avoir une différences de navigation en fonction des téléphones.

La connexion internet est également une contrainte pour l'application. Puisqu'elle récupère les données de la base de données, il est crucial qu'il existe une connexion. De plus, en fonction du réseau, il est possible d'avoir une différence du temps de réception des données plus lente. Il existera des fonctionnalités hors-ligne qui permettra entre autre de pouvoir basculer sur un nouveau protocole pour que les utilisateurs puissent continuer à communiquer entre eux sur le chat : le protocole SMS. L'ensemble des données des sessions récoltés sont alors stockées dans une base de données interne du mobile. Lors de la récupération du réseau internet, l'application va alors recevoir l'ensemble des données collectés en dur et va reprendre son état d'envoi naturel.

2.5 Hypothèses et dépendances

Le produit sera toujours utilisé sur les téléphones mobiles Android qui ont assez de performances. Si le téléphone ne dispose pas de ressources matérielles suffisantes pour l'application. Elle peut ne pas fonctionner comme prévu ou même pas du tout. De plus, la version minimum d'android doit être Jelly Belly (Version 16).

Une autre hypothèse est que les composants GPS de tous les téléphones fonctionnent de la même manière. Si les téléphone on un système GPS est différent des normes, l'application doit être spécifiquement adaptée à chaque interface.

2.6 Répartition des besoins

Dans le cas où le projet est retardé, certaines exigences pourraient être transférées au prochaine version de l'application. Les exigences avec une priorités haute doivent être développé lors du rendu de la première version de l'application.

Chapitre 3

Exigences particulières

Cette section contient toutes les exigences fonctionnelles et de qualité du système. Il donne une description du système et de toutes ses caractéristiques.

3.1 Exigences externe de l'interface

Cette section fournit une description détaillée de toutes les entrées et sorties du système. Il donne également une description des interfaces homme machine et fournit des prototypes de base de l'interface pour l'utilisateur.

3.2 Les interfaces utilisateurs

Un visiteur de l'application mobile devrait voir la page d'ouverture de session quand il ouvre l'application. Pour la connexion au compte utilisateur, il lui est absolument nécessaire de posséder un compte utilisateur composé d'une adresse mail et du mot de passe associé.

FIGURE 3.1 – Maquette de l'IHM pour la connexion

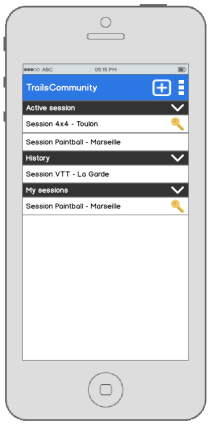
Pour la création d'un compte utilisateur, il est nécessaire de remplir l'ensemble des champs de l'interface. Toutefois, il sera possible de modifier son pseudonyme ainsi que son numéro de téléphone plus tard après sa connexion.

Après la connexion à un compte utilisateur, celui-ci peut alors choisir entre des sessions actives, les sessions dont il est l'organisateur ou des sessions clôturer qui sont dans la liste des historiques. L'icône plus permet de créer une nouvelle session. De plus, les boutons avec les 3 points permet d'afficher une liste déroulantes des différentes actions disponible sur cette interface. Les actions disponibles sont : la modification de ses données utilisateurs ainsi que la déconnexion.

Pour la modification des données utilisateurs, seulement le pseudonyme et le numéro de téléphone peuvent être modifié.

FIGURE 3.2 – Maquette de l'IHM pour la création d'un compte utilisateur

FIGURE 3.3 – Maquette de l'IHM pour l'affichage des différentes sessions



La création d'une session possède de nombreux champs texte obligatoire. Les adresses des lieux de départ et d'arrivé seront directement convertis en données GPS. De plus, lors du clique sur le champ texte de la date de départ, ue boite de dialogue apparaitra avec un calendrier spécifique à android. A noté que le champ du mot de passe ne sera connu que par l'organisateur de la session.

Lorsque l'utilisateur ouvre une session qui est dans sa liste de session historique. Cette interface apparait avec des actions propres au session historique.

Cette maquette montre l'interface d'une session complètement vide. Seulement la Google map y apparait. Lorsque l'utilisateur a choisit une session dont il est l'organisateur, une action est

FIGURE 3.4 – Maquette de l'IHM pour la modification des données utilisateurs

FIGURE 3.5 – Maquette de l'IHM pour la création d'une session

disponible lorsque celui-ci clique sur les trois petits points : clôturer la session. Une popup de confirmation va alors apparaitre pour qu'il puisse confirmer son action.

La maquette principal est l'affichage des différentes positions des participants de la session courante. Sur cette maquette nous pouvons aussi voir comment l'utilisateur peut accepter ou non un waypoint partagé par un autre participant.

La maquette suivante présente le chat d'une session courante. La liste des messages ce rafraichi en temps réel.

FIGURE 3.6 – Maquette de l'IHM pour l'affichage d'une session historique

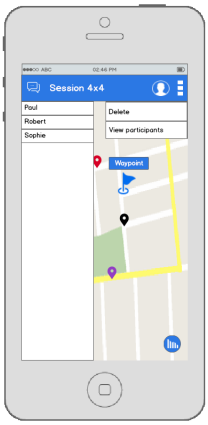
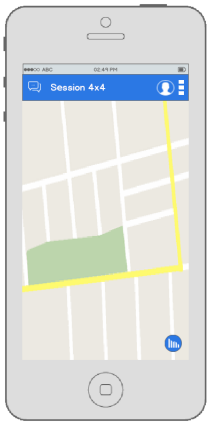


FIGURE 3.7 – Maquette de l'IHM pour l'affichage du Google map vide



Cette maquette montre l'affichage des statistiques du session courante. Les statistiques apparaissent lors du clique sur le bouton flottant en bas gauche.

Cette maquette montre l'interface pour que l'organisateur puisse modifier l'ensemble des données de sa session.

Cette maquette montre comment un waypoint est diffusé aux autres participants. L'utilisateur a alors le choix d'accepter ou de refuser que le waypoint diffusé apparaisse sur sa Google map.

FIGURE 3.8 – Maquette de l'IHM pour l'affichage des différentes positions des participants et de réponse au partage d'un waypoint.

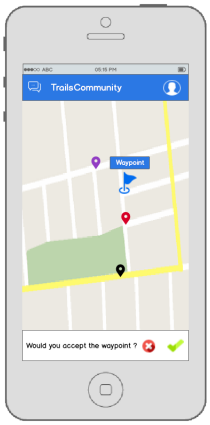
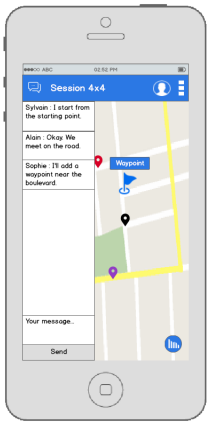


FIGURE 3.9 – Maquette de l'IHM pour le chat d'une session courante



3.3 Interfaces matérielles

Puisque l'application mobile ne comportent pas de matériel désigné, il n'y a pas d'interfaces matérielles directes. Le GPS physique est géré directement par l'application et la connexion matérielle au serveur est géré par le système d'exploitation sous-jacent sur le téléphone mobile.

FIGURE 3.10 – Maquette de l'IHM pour l'affichage des statistiques d'une session courante

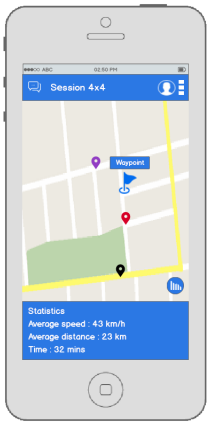
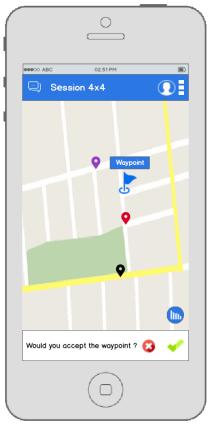


FIGURE 3.11 – Maquette de l'IHM pour la modification des données d'une session

3.4 Interfaces logicielles

L'application mobile communique avec le GPS pour obtenir les informations géographiques. Pour la localisation de l'utilisateur et la représentation visuelle de celle-ci, l'utilisation du serveur est primordiale afin d'obtenir en temps réel les informations sur les participants. La communication entre la base de données et l'application se compose des opérations d'écriture et de lecture.

FIGURE 3.12 – Maquette de l'IHM pour l'affichage de l'événement d'accepter un nouveau waypoint



3.5 Interfaces de communication

La communication entre les différentes parties du système est importante car elles dépendent les unes des autres. Cependant, la manière dont la communication est réalisée n'est pas importante pour le système et est dont traitée par les systèmes d'exploitation pour l'application mobile.y

Chapitre 4

Exigences fonctionnelles

Cette section comprend les exigences qui spécifient toutes les actions fondamentales du système logiciel.

4.1 Visiteur

4.1.1 Exigence fonctionnelle 1.1

ID : FR1

TITRE : Télécharger l'application mobile

DESCRIPTION : Un utilisateur doit être en mesure de télécharger l'application mobile via le Google play store. L'application doit être libre de téléchargement.

BUT : Pour qu'un utilisateur puisse télécharger l'application mobile.

DÉPENDANCE : Aucune.

4.1.2 Exigence fonctionnelle 1.2

ID : FR2

TITRE : Télécharger et notifier les utilisateurs des nouvelles versions.

DESCRIPTION : Lorsqu'une nouvelle version/mise à jour est publiée, l'utilisateur doit les vérifier manuellement. Le téléchargement de la nouvelle version devrait se faire via le téléphone mobile de la même manière que le téléchargement de l'application mobile.

BUT : Pour que l'utilisateur puisse télécharger une nouvelle version.

DÉPENDANCE : FR1.

4.1.3 Exigence fonctionnelle 1.3

ID : FR3

TITRE : Enregistrement du visiteur sur l'application mobile.

DESCRIPTION : Étant donné qu'un utilisateur a téléchargé l'application mobile, l'utilisateur devrait pouvoir s'enregistrer via l'application mobile. L'utilisateur doit fournir en premier lieu son surnom puis vient en deuxième, adresse mail ainsi que son mot de passe et pour finir son numéro de téléphone.

BUT : Pour qu'un visiteur puisse s'inscrire sur l'application mobile.
DÉPENDANCE : FR1.

4.1.4 Exigence fonctionnelle 1.4

ID : FR4
TITRE : Connexion du visiteur sur l'application mobile.
DESCRIPTION : Étant donné qu'un visiteur s'est enregistré, il doit pouvoir se connecter à l'application mobile via son adresse mail et son mot de passe. Les informations d'ouverture de session seront stockées dans la base de données.
BUT : Pour qu'un visiteur puisse se connecter sur l'application mobile.
DÉPENDANCE : FR1, FR3.

4.2 Utilisateur

4.2.1 Exigence fonctionnelle 2.1

ID : FR5
TITRE : Un utilisateur peut rejoindre une session.
DESCRIPTION : Étant donné qu'un utilisateur s'est connecté, il doit à le choix entre 3 types de sessions. Le premier type est les sessions actives, elles sont soit privée ou publique. Si elles sont privées, l'utilisateur doit connaitre le mot de passe pour les rejoindre. Ensuite, le deuxième type est les sessions dont il a déjà participé, cela fonctionne comme un historique. La dernière est les sessions que l'utilisateurs à créées.
BUT : Pour qu'un organisateur puisse rejoindre une session sur l'application mobile.
DÉPENDANCE : FR1, FR3, FR4.

4.2.2 Exigence fonctionnelle 2.2

ID : FR6
TITRE : Un utilisateur peut modifier ses données utilisateurs.
DESCRIPTION : Étant donné qu'un utilisateur s'est connecté, il peut modifier ses données utilisateurs. Les champs modifiables sont son pseudonyme, son numéro de téléphone. Les informations des données utilisateur seront stockées dans la base de données. Dans le cas ou celui-ci les modifie alors qu'il participe a une session. Les autres utilisateurs sont alors notifié si l'utilisateur modifie son pseudo. La notification apparaitra alors dans le chat.
BUT : Pour qu'un utilisateur puisse modifier ses données utilisateur sur l'application mobile.
DÉPENDANCE : FR1, FR3, FR4.

4.2.3 Exigence fonctionnelle 2.3

ID : FR7
TITRE : Un utilisateur peut se déconnecter de l'application.
DESCRIPTION : Étant donné qu'un utilisateur s'est connecté, il doit avoir le choix, à tout moment de pouvoir se déconnecter de l'application. Il sera alors redirigé vers la page de connexion.

BUT : Pour qu'un organisateur puisse se déconnecter de l'application mobile.
DÉPENDANCE : FR1, FR3.

4.3 Participant

4.3.1 Exigence fonctionnelle 3.1

ID : FR8

TITRE : Un participant peut ajouter un waypoint dans une session active.

DESCRIPTION : Étant donné qu'un participant s'est connecté et à rejoint une session active, il peut ajouter un waypoint. Lorsque l'utilisateur maintient longuement son doigt sur l'écran de son téléphone, un marker est alors ajouté. Le waypoint est alors diffusé à tout les autres participants. Les autres utilisateurs ont alors le choix d'accepter ou de décliner le partage.

BUT : Pour qu'un participant puisse ajouter un waypoint dans une session active.
DÉPENDANCE : FR1, FR3, FR4, FR5.

4.3.2 Exigence fonctionnelle 3.2

ID : FR9

TITRE : Un participant peut communiquer via un système de chat en temps réel à la session courante.

DESCRIPTION : Étant donné qu'un participant s'est connecté et à rejoint une session active, il peut alors communiquer avec l'ensemble des utilisateurs participant à la session. Il lui suffit d'entrer son message et de l'envoyer. Il sera alors diffusé aux autres participants. De plus, si l'application est hors-ligne. Les messages seront alors envoyé en protocole SMS et stocké dans a base interne pour qu'ils puissent être envoyé au serveur ultérieurement.

BUT : Pour qu'un participant puisse communiquer via un système de chat en temps réel à la session courante.
DÉPENDANCE : FR1, FR3, FR4, FR5.

4.3.3 Exigence fonctionnelle 3.3

ID : FR10

TITRE : Un participant peut visualiser ses statistiques de la session active.

DESCRIPTION : Étant donné qu'un participant s'est connecté et à rejoint une session active, il peut alors visualiser l'ensemble de ses statistiques. Celle-ci décrivent différentes données comme la distance total parcouru, la vitesse moyenne du participant ou alors le nombre de waypoint ajoutés.

BUT : Pour qu'un participant puisse visualiser ses statistiques de la session active
DÉPENDANCE : FR1, FR3, FR4, FR5.

4.4 Organisateur

4.4.1 Exigence fonctionnelle 4.1

ID : FR11

TITRE : Un organisateur peut créer une session.

DESCRIPTION : Étant donné qu'un organisateur s'est connecté, il doit pouvoir créer une nouvelle session dans l'application. Les informations de la nouvelle session créée seront stockées dans la base de données du serveur distant. L'organisateur doit fournir le lieu de départ ainsi que le lieu d'arrivé. Le format d'une adresse lui sera demandé. L'application convertira cette adresse en coordonnées GPS. De plus, il devra indiquer le type d'activité ainsi que la date de départ. Pour finir, l'organisateur aura la possibilité de verrouiller la session par un mot de passe qu'il divulguera à l'ensemble des participants.

BUT : Pour qu'un organisateur puisse créer une session sur l'application mobile.

DÉPENDANCE : FR1, FR3, FR4.

4.4.2 Exigence fonctionnelle 4.2

ID : FR12

TITRE : Un organisateur peut gérer ses sessions.

DESCRIPTION : Étant donné qu'un organisateur s'est connecté, il doit pouvoir gérer ses sessions. Touts les champs remplit précédemment lors de la création de la session sont modifiables. Les informations de la session seront stockées dans la base de données.

BUT : Pour qu'un organisateur puisse gérer une session sur l'application mobile.

DÉPENDANCE : FR1, FR3, FR4, FR5.

4.4.3 Exigence fonctionnelle 4.3

ID : FR13

TITRE : Un organisateur peut clôturer ses sessions.

DESCRIPTION : Étant donné qu'un organisateur s'est connectée à sélectionné une de ses session créée précédemment. Il doit pouvoir clôturer la session sélectionné. Celle-ci n'est alors plus présente dans les sessions actives de l'application sauf pour les participants ou la session est alors disponible dans leurs historique. Cette action est disponible dans le menu déroulant des trois petits points (voir maquette 3.7). Une popup de confirmation va alors apparaitre pour qu'il puisse confirmer ou annuler son action.

BUT : Pour qu'un organisateur puisse clôturer une session sur l'application mobile.

DÉPENDANCE : FR1, FR3, FR4, FR5.

4.5 Application

4.5.1 Exigence fonctionnelle 5.1

ID : FR14

TITRE : L'application doit pouvoir enregistrer un parcours effectué sans connexion internet.

DESCRIPTION : Étant donné qu'un utilisateur s'est connecté, à rejoint une session active et qu'il est déconnecté du réseau internet, l'application doit pouvoir gérer le mode hors ligne. Une base de données interne a l'application va alors stockées l'ensemble des données. Lorsque l'utilisateur sera de nouveau connecté, les informations seront alors synchronisé avec le serveur.

BUT : Pour que l'application puisse pouvoir enregistrer un parcours effectué sans connexion internet.

DÉPENDANCE : FR1, FR3, FR4, FR5.

4.5.2 Exigence fonctionnelle 5.2

ID : FR15

TITRE : Le chat de l'application doit pouvoir fonctionné en mode hors ligne.

DESCRIPTION : Étant donné qu'un utilisateur s'est connecté, à rejoint une session active et qu'il est déconnecté du réseau internet, l'application doit pouvoir gérer le mode hors ligne du chat. L'application va alors basculé sur le protocole SMS pour que les participants puissent continués de communiquer ensemble. De plus, l'ensemble des messages seront stocké en internet et synchronisé avec le serveur lorsque l'utilisateur sera de nouveau en ligne.

BUT : Pour que le chat de l'application puisse fonctionner en mode hors ligne.

DÉPENDANCE : FR1, FR3, FR4, FR5.

4.6 IHM

4.6.1 Exigence fonctionnelle 6.1

ID : FR16

TITRE : Le système doit pouvoir afficher une carte Google map interactive.

DESCRIPTION : Le système doit être capable d'afficher une carte Google map interactive. En effet, cette carte doit pouvoir posséder les fonctionnalités de zoom ainsi qu'une navigation tactile et ergonomique.

BUT : Pour que le système puisse afficher une carte Google map interactive.

DÉPENDANCE : Aucune.

4.6.2 Exigence fonctionnelle 6.2

ID : FR17

TITRE : Le système doit pouvoir ajouter un ou plusieurs markers sur une carte Google map.

DESCRIPTION : Étant donné que le système a affiché une carte Google map interactive, il doit permettre de pouvoir ajouter un ou plusieurs marker sur celle-ci. En effet ces markers peuvent posséder un titre interactif qui, lors d'une appuie, peuvent afficher une multitude d'informations supplémentaires sur le marker associé.

BUT : Pour que le système puisse ajouter un ou plusieurs markers sur une carte Google map.

DÉPENDANCE : F16.

4.6.3 Exigence fonctionnelle 6.3

ID : FR18

TITRE : Le système doit pouvoir afficher un ou plusieurs parcours sur une carte Google map.

DESCRIPTION : Étant donné que le système a affiché une carte Google map interactive, il doit permettre de pouvoir afficher un ou plusieurs parcours sur celle-ci. En effet ces chemins possède un

marker de début et un marker de fin pour signaler aux autres participants où ce situe l'utilisateur. Ce parcours possèdera une couleur unique dans la session active.

BUT : Pour que le système puisse afficher un ou plusieurs parcours sur une carte Google map.

DÉPENDANCE : F16, F17.

4.6.4 Exigence fonctionnelle 6.4

ID : FR19

TITRE : Le système doit pouvoir afficher la géo-localisation des participants d'une session active sur une carte Google map.

DESCRIPTION : Étant donné que le système a affiché une carte Google map interactive, il doit permettre de pouvoir afficher la géo-localisation des participants d'une session active sur celle-ci. En effet cette géo-localisation est signalé par un marker au bout du chemin associé aux participants. Cette géo-localisation est réalisé par le GPS interne du téléphone portable. De plus, la précision de la celle-ci dépendra totalement du système d'exploitation et du matériel du téléphone portable de l'utilisateur.

BUT : Pour que le système puisse afficher la géo-localisation des participants d'une session active sur une carte Google map.

DÉPENDANCE : F16.

4.6.5 Exigence fonctionnelle 6.5

ID : FR20

TITRE : Le système doit pouvoir afficher différentes unités de mesures des statistiques en fonction des activités.

DESCRIPTION : Le système doit pouvoir afficher différentes unités de mesures en fonction du type d'activité. En effet, les mesures sont différentes pour une activité ce réalisant avec une voiture et une activité ce réalisant en bateau ou à pied. Le système va alors déterminer en fonction du type d'activité choisit l'unité de mesure le plus adéquate.

BUT : Pour que le système puisse afficher différentes unités de mesures des statistiques en fonction des activités.

DÉPENDANCE : Aucune.

4.6.6 Exigence fonctionnelle 6.6

ID : FR21

TITRE : Le système doit pouvoir gérer la gestion un mode multi-session.

DESCRIPTION : Le système doit permettre un système de multi-session. Il permettra a l'utilisateur de pouvoir utiliser son compte sur différent mobile.

BUT : Pour que le système puisse gérer un mode multi-session.

DÉPENDANCE : Aucune.

4.7 Performances

4.7.1 Exigence fonctionnelle 7.1

ID : FR22

TITRE : Le système doit afficher rapidement l'ensemble des sessions dans les listes déroulantes.

DESCRIPTION : Le système doit posséder un niveau de performance élevé pour l'affichage des différentes sessions envoyé par le serveur. En effet, la rapidité d'affichage doit être rapide et aucune latence ne doit être observé par l'utilisateur. De plus, un maximum de 50 sessions actives sera affichées.

BUT : Pour que le système puisse afficher rapidement l'ensemble des sessions dans les listes déroulantes.

DÉPENDANCE : Aucune.

4.7.2 Exigence fonctionnelle 7.2

ID : FR23

TITRE : Le système doit afficher rapidement l'ensemble des différents éléments afficher sur la carte Google map interactive.

DESCRIPTION : Le système doit posséder un niveau de performance élevé pour l'affichage des différents éléments afficher sur la carte Google map. En effet, aucune latence ne doit être observé par l'utilisateur lors de la réception de coordonnées GPS ou l'affichage d'un nouveau waypoint. De plus, l'affichage des parcours de l'ensemble des participants d'une session active ne doit pas entrainé une surcharge du serveur.

BUT : Pour que le système puisse afficher rapidement l'ensemble des différents éléments afficher sur la carte Google map interactive.

DÉPENDANCE : Aucune.

4.7.3 Exigence fonctionnelle 7.3

ID : FR24

TITRE : Le serveur doit posséder un temps de réponse inférieur à une seconde.

DESCRIPTION : Le serveur doit posséder un temps de réponse inférieur à une seconde. Ce temps est un temps de performance idéal cependant, le temps de réponse doit être au minimum inférieur à deux secondes.

BUT : Pour que le Le serveur doit posséder un temps de réponse inférieur à une seconde.

DÉPENDANCE : Aucune.

4.8 Sécurité

4.8.1 Exigence fonctionnelle 7.1

ID : FR25

ESSENTIEL : La sécurité de création de compte pour les utilisateurs du système.

DESCRIPTION : Si un utilisateur veut créer un compte et que le nom de l'utilisateur souhaité est déjà utilisé, l'utilisateur va être invité à choisir un nom d'utilisateur différent.

NIVEAU : Haute.

4.8.2 Exigence fonctionnelle 7.2

ID : FR26

ESSENTIEL : La sécurité de la communication entre le système et le serveur.

DESCRIPTION : Les messages doivent être chiffrés pour les communications de connexion, de sorte que d'autres utilisateurs ne peuvent pas obtenir le nom d'utilisateur et le mot de passe à partir de ces messages.

NIVEAU : Haute.

4.8.3 Exigence fonctionnelle 7.3

ID : FR27

ESSENTIEL : Le nombre de requête par seconde côté serveur.

DESCRIPTION : Le serveur peut ajouter une limitation de requête par seconde pour un utilisateur. Cela permet d'évité les attaques de brute forces et autre attaques d'injection de code.

NIVEAU : Haute.

4.8.4 Exigence fonctionnelle 7.4

ID : FR28

ESSENTIEL : Utilisation d'un token pour la communication client-serveur.

DESCRIPTION : Les échanges entre client-serveur se fait grâce à un token générer par le serveur. Ce token est unique et propre à chaque utilisateur. L'utilisation de ce token permet de sécuriser les ressources du serveur. A chaque requête, le token doit être présent dans l'en-tête. S'il n'y a pas de token, le serveur renvoie une réponse comme quoi l'utilisateur n'est pas autorisé à accéder à cette ressource.

NIVEAU : Haute.

4.9 Test

4.9.1 Test unitaire

Identifiant	Titre	Description

4.9.2 Test UI

Identifiant	Titre	Description

Chapitre 5

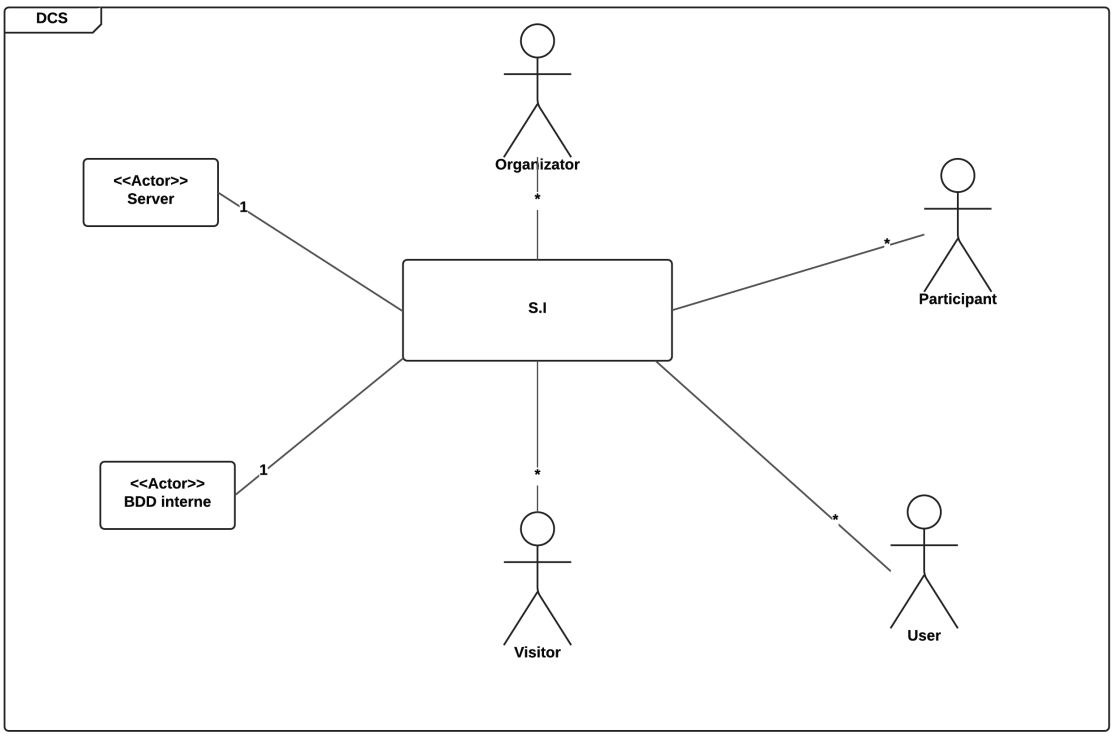
Conception UML

5.1 Modélisation de l'axe statique

5.1.1 Diagramme de contexte statique

Nous distinguons quatres types d'acteurs : un visiteur, un utilisateur, un participant et un organisateur. Chaque acteurs auront des utilisations de l'application mobile complètement distincts.

FIGURE 5.1 – Diagramme de contexte statique



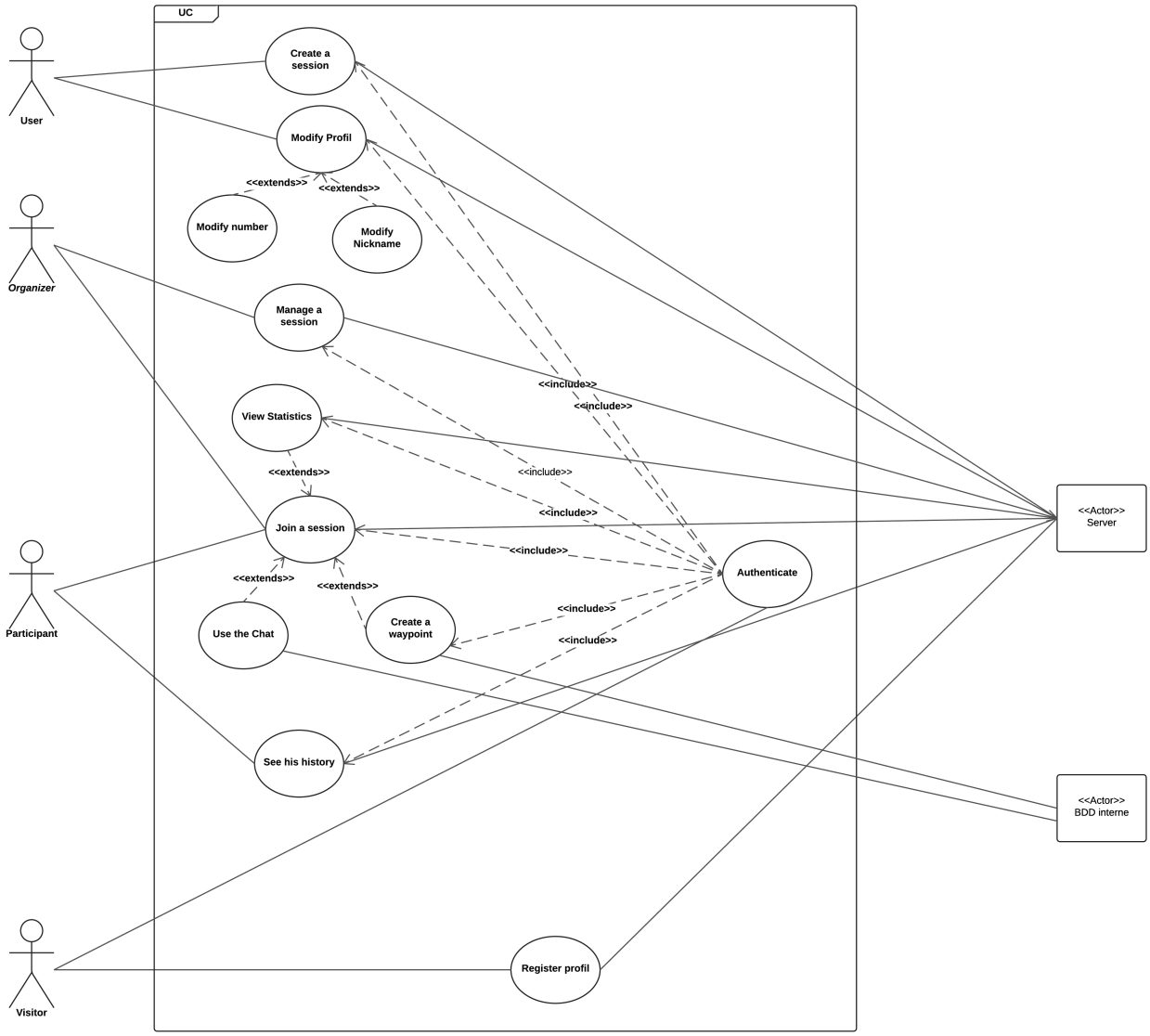
L'accès aux données nécessaires au fonctionnement de l'application se fera par le biais d'un serveur codé en Ruby dont le fonctionnement est détaillé en annexe. C'est ce serveur qui communiquera

avec la base de données qui stocke l'ensemble des informations des sessions.

5.1.2 Diagramme des cas d'utilisation

L'ensemble des cas d'usages excepté la création d'un compte utilisateur requiere une authentification. Le statut de l'acteur diffère en fonction des actions qu'il réalise sur l'application.

FIGURE 5.2 – Diagramme des cas d'utilisation

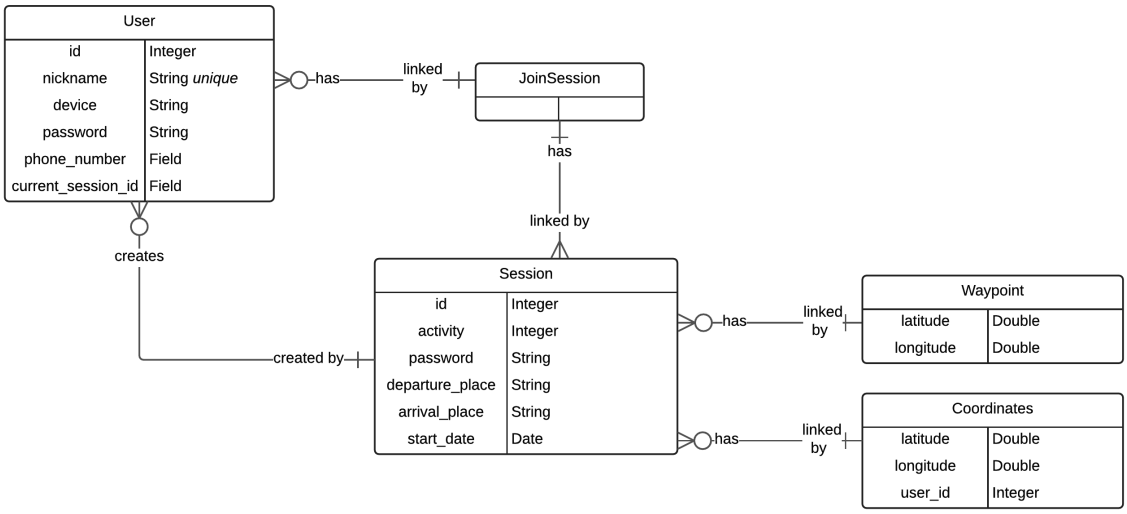


La priorité des cas d'utilisation est détaillée dans la figure ... La principal fonctionnalité de l'application est d'assurer la possibilité de créer une session d'une activité quelconque et d'y participer.

Use case	Priority	Risk
Create a session	High	High
Manage a session	High	High
Join a session	High	Mean
Create a waypoint	Mean	Mean
Authentification	Mean	Mean
Use the chat	Low	Low
See statistics	Low	Low
See his history	Low	Low
Modify profil	Low	Low

5.1.3 Diagramme de classe

FIGURE 5.3 – Diagramme entité association



5.2 Modélisation des axes fonctionnel et dynamique

5.2.1 Généralités

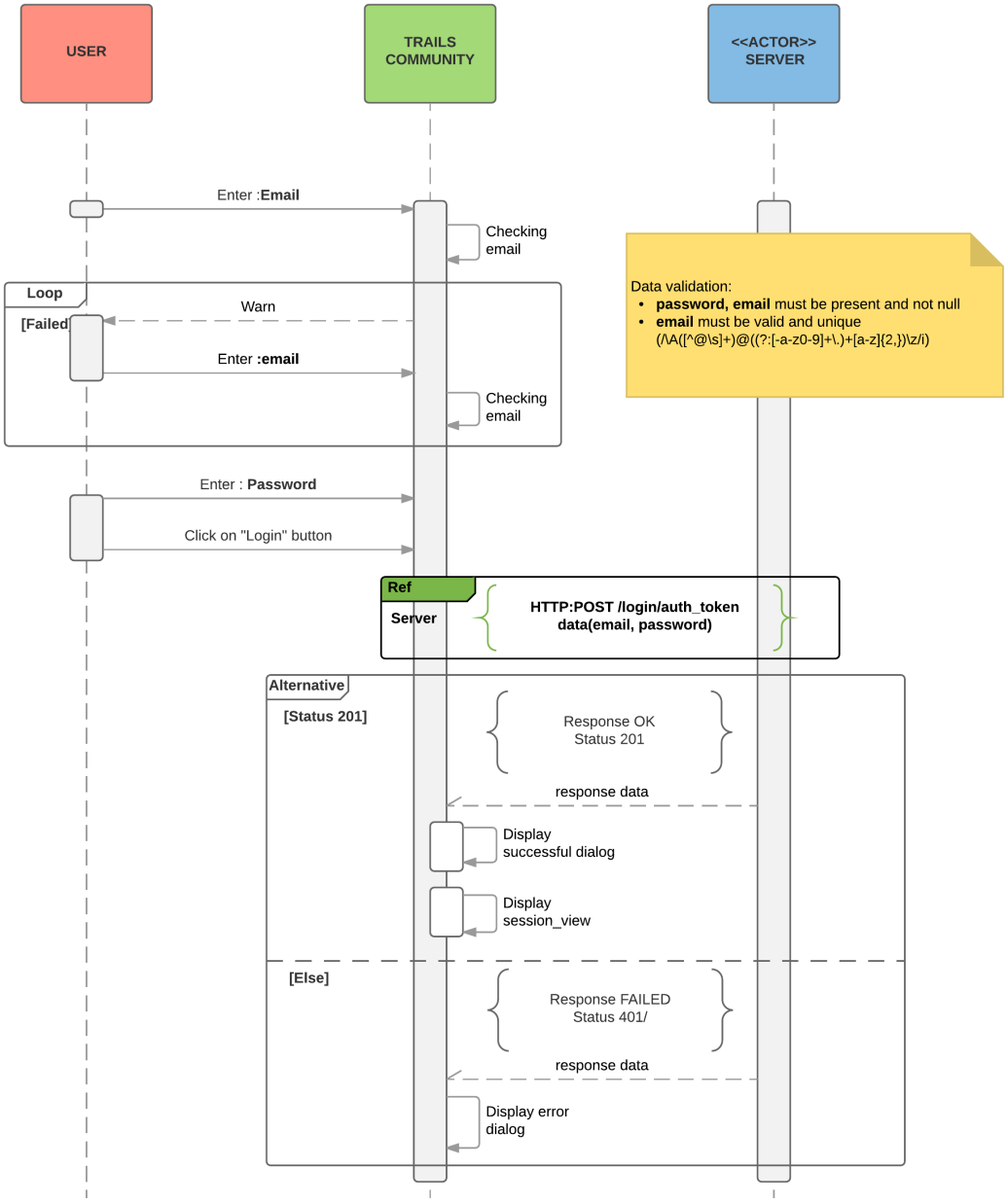
Tous les diagrammes de séquences et d'activité présentés ci-dessous seront depuis le point de vue du client.

Une rapide modélisation du serveur sera présente dans la partie 'Modélisation du serveur'. Les diagrammes de séquence comprennent cependant une redirection vers la réception ou l'envoi de données vers ou depuis le serveur.

Tous les cas d'utilisation débutent nécessairement par une connexion de l'utilisateur, décrite dans le diagramme de séquence en figure ...

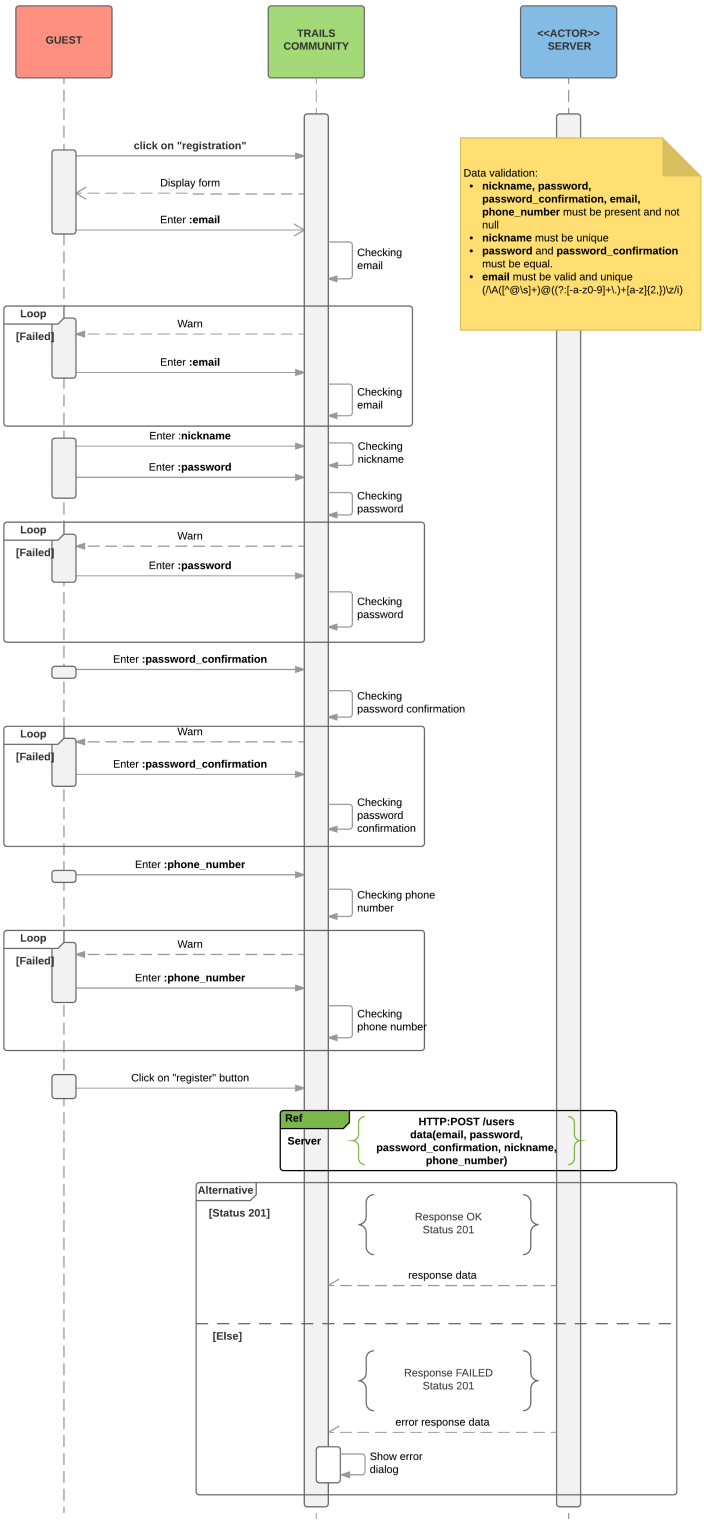
De plus l'ensemble des données sont testées sur le client mais aussi sur le serveur qui contient la base de données.

FIGURE 5.4 – Diagramme de séquence authentification



Avant sa première connexion, l'utilisateur devra créer un compte pour obtenir ses informations de connexion. Cette étape est primordiale car certaines données sont sensible car elle seront utilisés directement par l'application comme le pseudonyme ou le numéro de téléphone.

FIGURE 5.5 – Diagramme de séquence créer un compte utilisateur

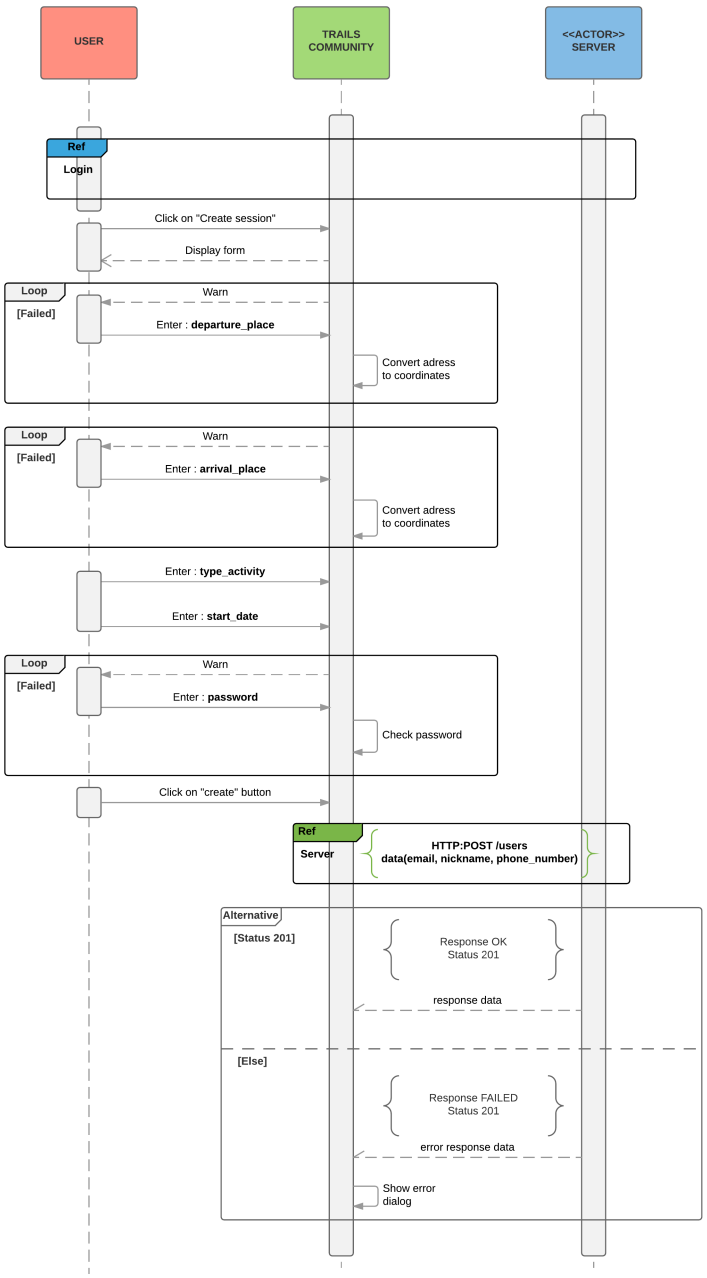


5.2.2 Traitement d'un cas d'utilisation - Créer une session

Le cas d'utilisation que nous avons choisi de détailler ici est la création d'une session. C'est un des cas d'utilisation essentiel au fonctionnement de notre application mobile. Sans cette étape, il est impossible de réaliser n'importe quelles actions concretes sur l'application.

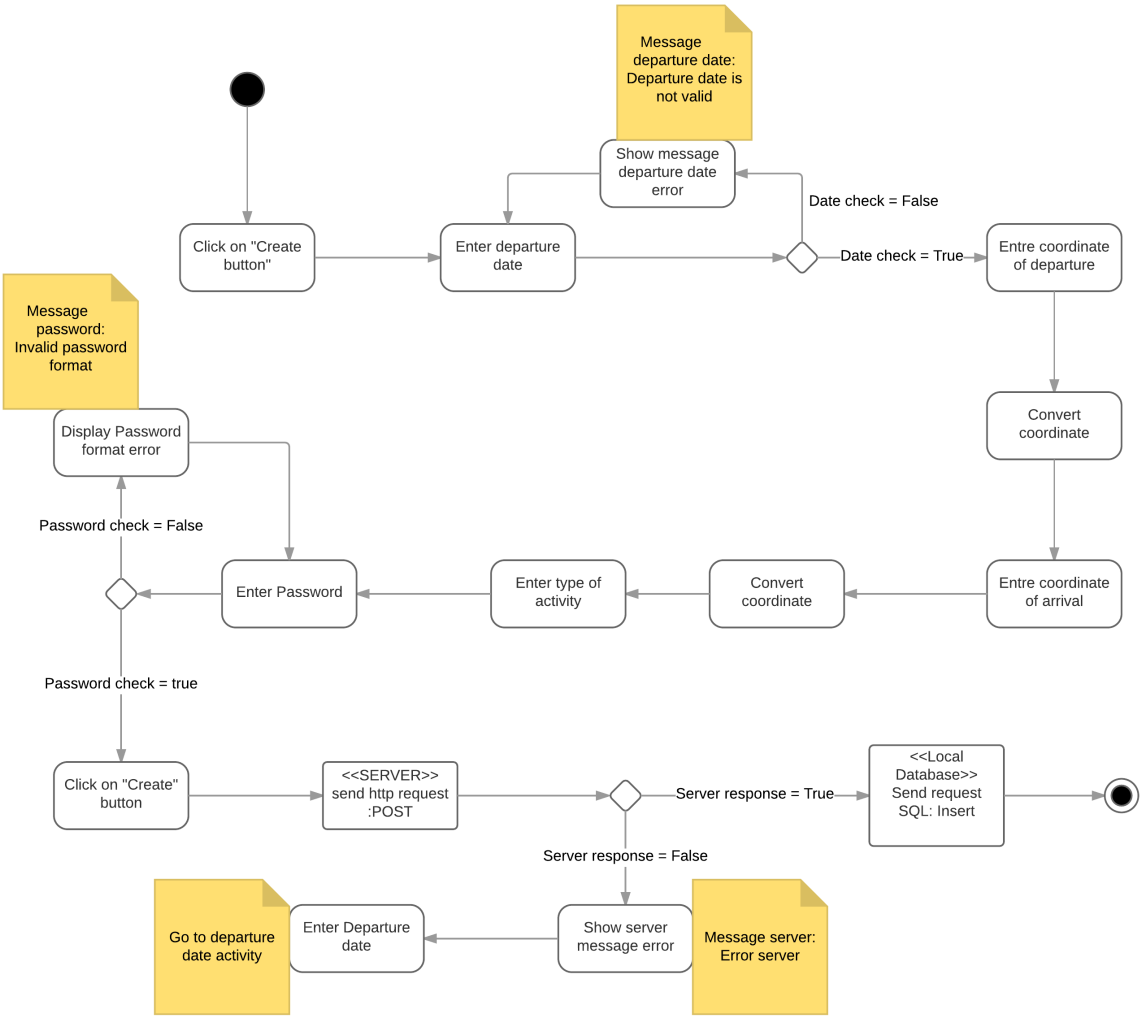
Le diagramme de séquence s'articule principalement autour de trois étapes : l'authentification, la saisie des données du sondage puis l'envoi des données au serveur.

FIGURE 5.6 – Diagramme de séquence créer une session



Comme on peut le voir sur le diagramme d'activité. La gestion d'erreur ce passe sur les les 2 applications : le modele et le serveur. De plus, le serveur se charge de remonter les erreurs sur le format des données ou la lecture/écriture en base de données. Le client informe l'utilisateur sur l'échec du format des données entré en paramètre et aussi l'échec des opération serveur.

FIGURE 5.7 – Diagramme d'activité de la création d'une session



Les cas d'utilisation de modification d'une session qui comprend la clôture d'une session est particulièrement similaire à ce cas d'utilisation.

5.2.3 Traitement d'un cas d'utilisation - Rejoindre une session

FIGURE 5.8 – Diagramme de séquence rejoindre une session

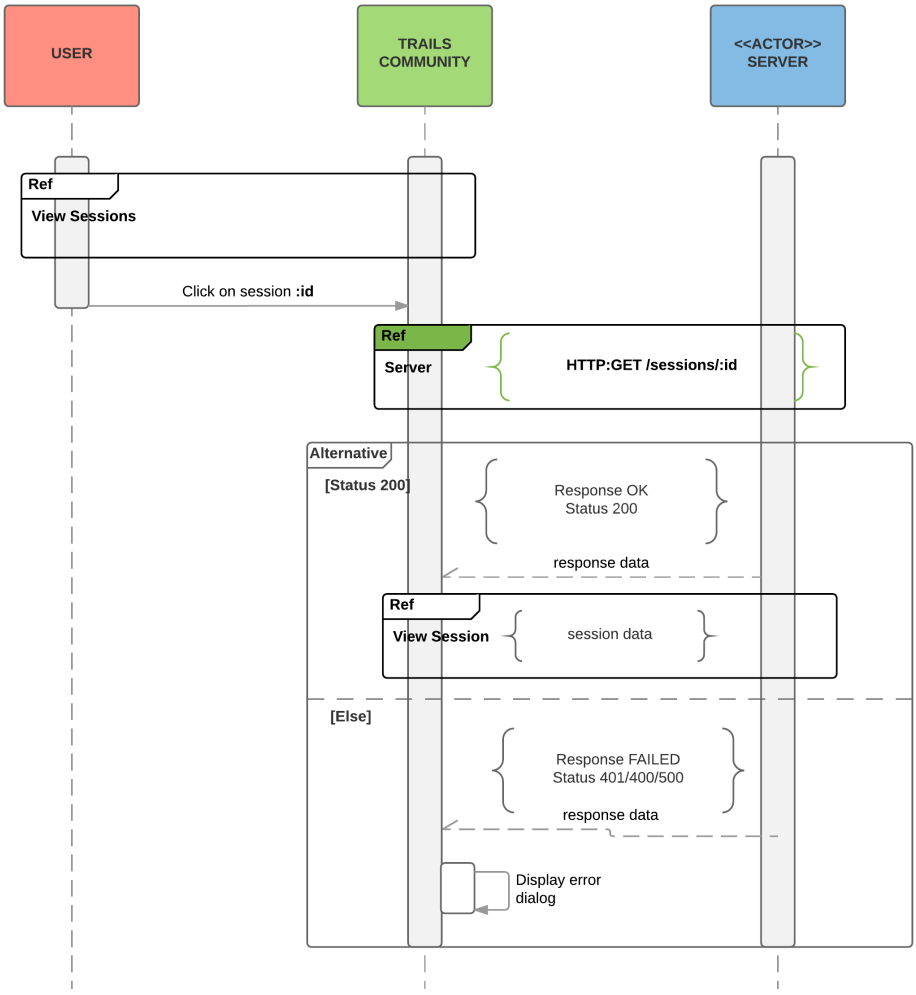


FIGURE 5.9 – Diagramme d'activité rejoindre une session

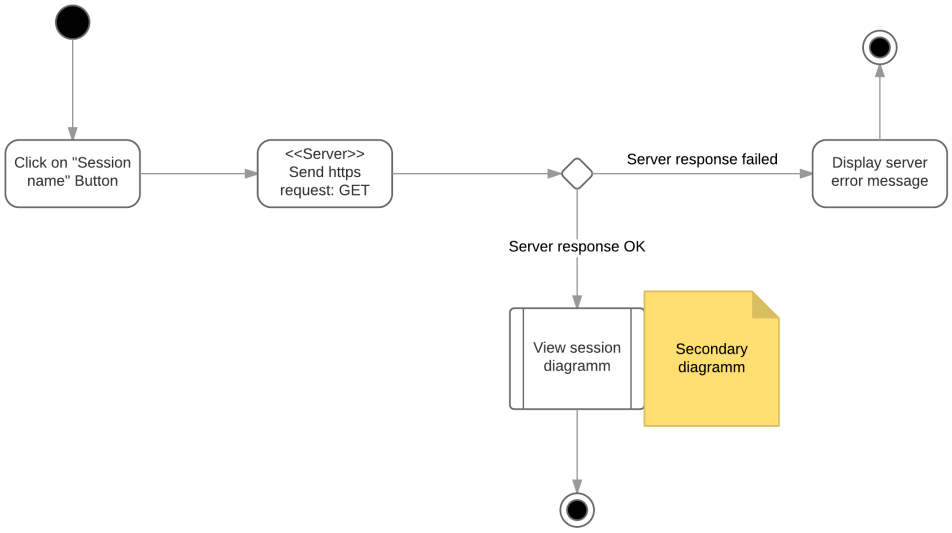


FIGURE 5.10 – Diagramme de séquence de l'affichage d'une session

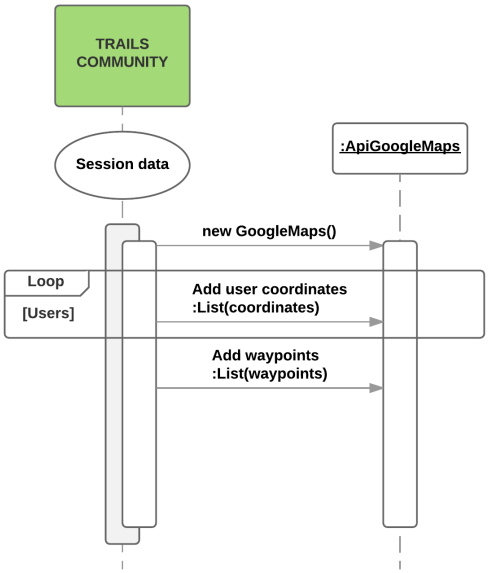
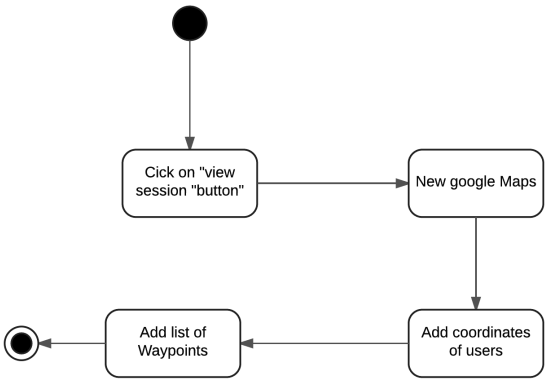


FIGURE 5.11 – Diagramme d'activité de l'affichage d'une session



Deuxième partie

Serveur TrailsCommunity

Chapitre 1

Introduction

1.1 Conventions

— **Client** : Application client.
— **Statut** : Code réponse de la requête HTTP
— Toutes les réponses possibles sont listés sous la catégorie 'Réponse'
— Toutes les réponses sont du format JSON

1.2 Codes statut

Tous les codes de statut sont des standards HTTP. Cette API utilisera les suivants :
— $2XX$ - Succès de la requête
— $4XX$ - Erreur du côté client
— $5XX$ - Erreur du côté serveur

Code statut	Description
200	Requête traitée avec succès
201	Requête traitée avec succès et création d'un document
202	Requête traitée avec succès mais pas d'information à renvoyer
400	La syntaxe de la requête est erronée
401	Une authentification est nécessaire pour accéder à la ressource
404	Ressource non trouvée
500	Erreur interne du serveur
503	Service temporairement indisponible ou en maintenance

Chapitre 2

Ressources

2.1 Authentification

2.1.1 Requête

Méthode	URL
POST	`api/login/auth_token`

Type	Paramètres	Valeurs
HEADER	`Application/json`	*String*
JSON	`{ "auth": { "email": "mail@mail.com", "password": "FZ,f235nDE", } }`	*String* *String*

email : correspond à l'adresse mail valide de l'utilisateur
password : mot de passe de l'utilisateur

2.1.2 Réponse

Statut	Réponse
200	`{ "jwt": <Authorization> }`
401	
500	

Authorization(*string*) - Tous les autres appels nécessitant une authentification doivent avoir cette clé dans le header.

2.2 Inscription

2.2.1 Requête

Enregistrement d'un utilisateur dans le système.

Méthode	URL
POST	api/users/

Type	Paramètres	Valeurs
HEADER	Application/json	*String*

JSON	{
	"nickname": "Arthur83",
	"email": "apaul@custom.com",
	"phone_number": "+33651678908",
	"password": "myPassword33",
	"password_confirmation": "myPassword33"
	}
		
		String
		String
		String
		String
		String

nickname : pseudonyme de l'utilisateur
email : email de l'utilisateur non nul et valide
password : mot de passe d'utilisateur
$password_confirmation$: confirmation du mot de passe de l'utilisateur. Il doit être identique à *password*.

2.2.2 Réponse

Statut	Réponse

201	{
	"data": {
	"id": 2,
	"nickname": "Arthur83",
	"phone_number": "+33651678908",
	"current_session_id": null
	}
	}

400	{
	"errors": {
	"nickname": [
	"has already been taken"
	],
	"email": [
	"is invalid"
	]
	}
	}

500	

2.3 Ajout d'une session

2.3.1 Requête

Permet d'ajouter une nouvelle session dans le serveur. Cette ressource est contrainte d'un token d'authentification *<Authorization>* qui a été donnée avec la ressource *api/login/auth_token*

Méthode	URL
POST	api/sessions/

Type	Paramètres	Valeurs
HEADER	Application/json	*String*
HEADER	Authorization	*String*

JSON	{
	"activity": 1,
	"password": "lockSessionPsw",
	"departure_place": "43.179363;5.717782",
	"arrival_place": "43.191168;5.730819",
	"start_date": "2016-11-20"
	}
		Integer
		String
		String
		String
		Date

activity : identifiant de l'activité. (1 : randonnée, 2 : vélo, ...)
password : mot de passe de la session, permet de restreindre l'entrée.
departure_place : lieu de départ. Couple de position géographique (*latitude*; *longitude*)
arrival_place : lieu d'arrivée. Couple de position géographique (*latitude*; *longitude*)

2.3.2 Réponse

Statut	Réponse

200	{
	"data": {
	"id": 5,
	"activity": 1,
	"departure_place": "43.179363;5.717782",
	"arrival_place": "43.191168;5.730819",
	"start_date": "2016-11-20",
	"close": false,
	"lock": true,
	"user": {
	"id": 1,
	"nickname": "Raptor234",
	"phone_number": "+33623569450",
	"current_session_id": null
	},
	"coordinates": [],
	"waypoints": []
	}
	}

400	{
	"errors": {
	"activity": [
	"can't be blank",
	"is not a number"
	]
	}
	}

401	
500	

2.4 Liste des sessions

2.4.1 Requête

Permet de récupérer la liste des sessions du système. Cette ressource est contrainte d'un token d'authentification <*Authorization*> qui a été donnée avec la ressource *api/login/auth_token*

Méthode	URL
GET	api/sessions/

Type	Paramètres	Valeurs
HEADER	Authorization	*String*

2.4.2 Réponse

Statut	Réponse

200	{
	"data": [
	{
	"id": 1,
	"activity": 1,
	"departure_place": "43.179363;5.717782",
	"arrival_place": "43.191168;5.730819",
	"start_date": "2016-11-20",
	"close": false,
	"lock": true,
	"user": {
	"id": 2,
	"nickname": "Arthur83",
	"phone_number": "+33651678908",
	"current_session_id": null
	}
	},
	{
	"id": 2,
	"activity": 3,
	"departure_place": "43.179363;5.717782",
	"arrival_place": "43.191168;5.730819",
	"start_date": "2016-11-20",
	"close": false,
	"lock": true,
	"user": {
	"id": 2,
	"nickname": "Arthur83",
	"phone_number": "+33651678908",
	"current_session_id": null
	}
	}
	]
	}

401	
500	

2.5 Session

2.5.1 Requête

Permet de récupérer une session spécifique selon son identifiant (ID). Cette ressource est contrainte d'un token d'authentification $<Authorization>$ qui a été donnée avec la ressource $api/login/auth\_token$

Méthode	URL
GET	api/sessions/:id

Type	Paramètres	Valeurs
URL_PARAM	id	*Integer*
$HEADER$	Authorization	*String*

id : identifiant de la session voulu

2.5.2 Réponse

Statut	Réponse

200	{
	"data": {
	"id": 2,
	"activity": 3,
	"departure_place": "43.179363;5.717782",
	"arrival_place": "43.191168;5.730819",
	"start_date": "2016-11-20",
	"close": false,
	"lock": true,
	"user": {
	"id": 2,
	"nickname": "Arthur83",
	"phone_number": "+33651678908",
	"current_session_id": null
	},
	"coordinates": [
	{
	"latitude": 43.179363,
	"longitude": 5.717782,
	"user_id": 1
	}
	],
	"waypoints": [
	{
	"latitude": 43.179363,
	"longitude": 5.717782
	}
	]
	}
	}

401	

404	{
	"errors": {
	"session": "Record Not Found"
	}
	}

500	

2.6 Mis à jour d'une session

2.6.1 Requête

Permet de mettre à jour session spécifique selon un identifiant (ID) dans le système. Cette ressource est contrainte d'un token d'authentification <*Authorization*> qui a été donnée avec la ressource *api/login/auth_token*

Méthode	URL
PUT	api/sessions/:id

Type	Paramètres	Valeurs
HEADER	Application/json	*String*
HEADER	Authorization	*String*
JSON	{ "close": true }	*boolean*

close : boolean determinant si la session doit être fermée.

2.6.2 Réponse

Statut	Réponse
200	{ "data": { "id": 7, "activity": 3, "departure_place": "43.179363;5.717782", "arrival_place": "43.191168;5.730819", "start_date": "2016-11-20", "close": true, "lock": true, "user": { "id": 2, "nickname": "Arthur83", "phone_number": "+33651678908", "current_session_id": 1 }, "coordinates": [], "waypoints": [] } }
401	
500	

2.7 Rejoindre une session

2.7.1 Requête

Permet de à l'utilisateur de rejoindre une session spécifique selon son identifiant (ID). Cette ressource est contrainte d'un token d'authentification <*Authorization*> qui a été donnée avec la ressource *api/login/auth_token*. De plus, un mot de passe peut être demandé si la session est restreinte.

Méthode	URL
GET	`api/sessions/:id/join`

Type	Paramètres	Valeurs
HEADER	`Authorization`	*String*
URL_PARAM	`id`	*Integer*
URL_PARAM	`password (optional)`	*String*

id : identifiant de la session voulu

2.7.2 Réponse

Statut	Réponse
200	
400	`{ "errors": { "session": "Bad password." OR "session": "This session needs a password to join it." } }`
401	
404	`{ "errors": { "session": "Record Not Found" } }`
500	

2.8 Ajout d'un waypoint

2.8.1 Requête

Permet de à l'utilisateur d'ajouter une nouveau waypoint pour une session spécifique selon son identifiant (ID). Cette ressource est contrainte d'un token d'authentification <*Authorization*> qui a été donnée avec la ressource *api/login/auth_token*

Méthode	URL
POST	api/sessions/:id/waypoint

Type	Paramètres	Valeurs
URL_PARAM	id	*Integer*
HEADER	Authorization	*String*

JSON	{
	"latitude": 43.179363,
	"longitude": 5.717782
	}
		Double
		Double

id : identifiant de la session voulu

2.8.2 Réponse

Statut	Réponse
200	
401	

404	{
	"errors": {
	"session": "Record Not Found"
	}
	}

500	